(19) Eur päisches Patentamt
Eur pean Patent Office
Office européen des brevets

(11) EP 1 041 496 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
04.10.2000 Bulletin 2000/40

(51) Int. Cl.[7]: **G06F 17/30**

(21) Application number: **00302491.6**

(22) Date of filing: **27.03.2000**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: **31.03.1999 US 282692**

(71) Applicant:
International Business Machines Corporation
Armonk, NY 10504 (US)

(72) Inventor:
Lita, Christian,
c/o IBM United Kingdom Ltd.
Winchester, Hampshire S021 2JN (GB)

(74) Representative:
Burt, Roger James, Dr.
IBM United Kingdom Limited
Intellectual Property Department
Hursley Park
Winchester Hampshire SO21 2JN (GB)

(54) Using virtual URLs for load balancing

(57) A method, computer program product and server for use managing connection requests to a pool of servers identified by a given URL. The method begins in response to a connection request from a given client machine that initiates a user session for associating a session identifier with a given server in the pool. The session identifier is then used to generate a "virtual" URL that redirects the connection request to the given server. Thereafter, any additional connection requests issued from the given client machine during the user session are redirected to the given server so that all content is served to the client from the same location. When the user session terminates, the virtual URL is inactivated and the given server is returned to the pool so that it can then be assigned a new user session to manage.

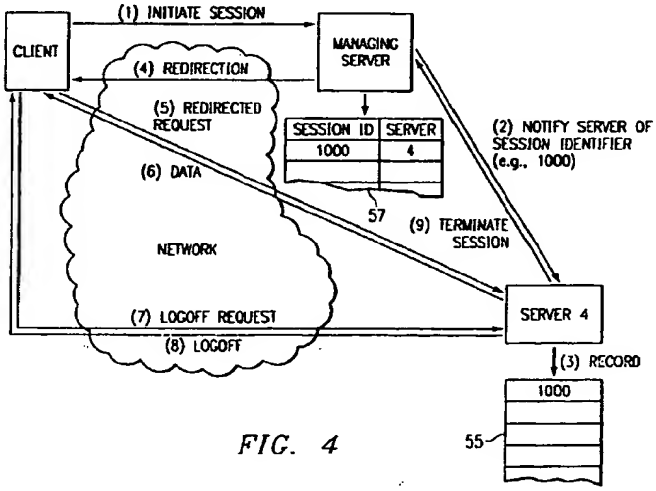

FIG. 4

EP 1 041 496 A2

Description

[0001] This invention relates generally to information retrieval in a computer network. More particularly, the invention relates to a method and system for balancing HTTP requests to a set of servers on a per-session as opposed to per-connection basis.

[0002] The World Wide Web is the Internet's multimedia information retrieval system. In the Web environment, client machines effect transactions to Web servers using the Hypertext Transfer Protocol (HTTP), which is a known application protocol providing users access to files (e.g., text, graphics, images, sound, video, etc.) using a standard page description language known as Hypertext Markup Language (HTML). HTML provides basic document formatting and allows the developer to specify "links" to other servers and files. In the Internet paradigm, a network path to a server is identified by a so-called Uniform Resource Locator (URL) having a special syntax for defining a network connection. Use of an HTML-compatible browser (e.g., Netscape Navigator or Microsoft Internet Explorer) at a client machine involves specification of a link via the URL. In response, the client makes a request to the server identified in the link and, in return, receives in return a document or other object formatted according to HTML. A collection of documents supported on a Web server is sometimes referred to as a Web site.

[0003] At many popular Web sites, the capacity demand is much greater than can be served by one server. Thus, it is known in the art to mirror a Web site and to incorporate a load balancing routine to distribute connection requests across a pool of servers. Typically, such load balancing is carried out in a round-robin fashion. Each server preferably includes the same data, so any request can be handled by any of the multiple servers in the pool. This distributes the load in an even manner.

[0004] While such known load balancing schemes are advantageous, certain types of Web server transactions are not conducive to load balancing in this manner. Thus, for example, assume that the Web site is a financial institution or bank. Typically, a user accesses such a site to effect a set of one or more transactions, e.g., an account balance inquiry, a transfer of funds between a set of pair of accounts, and the like. When this type of site is mirrored for load balancing purposes, there is a likelihood that the user's HTTP requests will be serviced by different servers in the server pool. This is undesirable, and it may force the user to have to log-in repeatedly to complete all of the transactions required. Similar problems are encountered in mirrored sites that offer electronic commerce transactions.

[0005] This problem arises because prior art load balancing techniques operate on a per-connection basis. Thus, if a given server handling an HTTP request suddenly becomes overloaded due to an excessive number of connections, the load balancing routine automatically delivers a next connection request to another server in the pool, even if the request originates from the same client machine.

[0006] The present invention accordingly provides, in a first aspect, a method for managing connection requests to a pool of servers identified by a given URL, comprising the steps of: in response to a connection request from a given client machine that initiates a session, associating a session identifier with a given server in the pool; using the session identifier to redirect the connection request to the given server; and during the session, redirecting to the given server any additional connection requests from the given client machine.

[0007] The step of using the session identifier preferably includes generating a virtual URL. The virtual URL preferably comprises a URL in the connection request modified to include the session identifier.

[0008] The session identifier is preferably incorporated in data returned from the given server to the client machine.

[0009] The method according to the first aspect preferably further includes the step of: in response to a connection request from the given client machine that terminates the session, inactivating the session identifier.

[0010] The given client machine preferably includes a browser.

[0011] Each of the servers in the pool preferably supports a similar set of objects.

[0012] The session identifier is preferably associated with a given server as a function of a load balancing protocol.

[0013] In a second aspect, the present invention provides a method for managing connection requests to a pool of servers, comprising the steps of: responsive to connection requests from client machines that initiate user sessions, associating each user session originating from a client machine with a given server in the pool in accordance with a load balancing protocol; and during the user session, redirecting to the given server any additional connection requests originating from the client machine.

[0014] The associating step preferably comprises: generating a virtual URL by modifying a given URL to include a session identifier; and using the virtual URL to redirect the connection request to the given server.

[0015] A method according to the second aspect preferably further includes the step of: inactivating the virtual URL upon completion of the user session.

[0016] All data returned from given server to the client machine preferably includes the session identifier.

[0017] Each of the servers in the pool preferably supports a similar set of given objects.

[0018] Each client machine preferably includes a Web browser.

[0019] In a third aspect, the present invention provides a computer program product in a computer-readable medium for managing connections requests to a

pool of servers, comprising: means responsive to connection requests from client machines that initiate user sessions for associating each user session originating from a client machine with a given server in the pool in accordance with a load balancing protocol; and means operative during each user session for redirecting to the given server any additional connection requests originating from the client machine.

[0020] The associating means preferably comprises: means for generating a virtual URL by modifying a given URL to include a session identifier; and means for redirecting a given connection request to the given server using the virtual URL.

[0021] A computer program product according to the third aspect preferably further includes means for inactivating the virtual URL upon completion of the user session.

[0022] In a fourth aspect, the present invention provides a server for managing a pool of servers at a Web site identified by a given URL, comprising: a processor; an operating system; a load balancing routine; and a redirector routine for managing HTTP connection requests to the Web site, comprising: means responsive to connection requests from client machines that initiate user sessions for associating each user session originating from a client machine with a given server in the pool in accordance with the load balancing routine; and means operative during each user session for redirecting to the given server any additional connection requests originating from the client machine.

[0023] The means for associating preferably comprises: means for generating a virtual URL by modifying the given URL to include a session identifier; and means for redirecting a given connection request to the given server using the virtual URL.

[0024] In a server according to the fourth aspect, the redirector preferably further includes means for inactivating the virtual URL upon completion of the user session.

[0025] In a fifth aspect, the present invention provides a method of managing a pool of servers at a Web site identified by a given URL, comprising the steps of: responsive to connection requests from client machines that initiate user sessions, associating each user session originating from a client machine with a server in the pool; distributing the user sessions across the pool of servers according to a load balancing protocol; and during a given user session initiated from a given client machine, serving content to the given client machine only from its associated server.

[0026] In a sixth aspect, the present invention provides a method of managing a pool of servers at a Web site identified by a given URL, comprising the steps of: during each user session initiated from a given client machine, temporarily redirecting all connection requests originating from the client machine to a given server in the pool; and distributing the user sessions across the pool of servers according to a load balancing protocol.

[0027] It is preferred in this invention to provide a method for equitably distributing client requests across a set of servers on a per-session basis. Preferably, a given server in the set is allocated a given number of sessions, as opposed to a given number of connections, and thus a user's HTTP connection requests are preferably serviced from the same server in the set throughout the session.

[0028] It is further preferred in this invention to implement a load balancing routine across a set of servers wherein given connection requests originating from a client machine are serviced from the same server in the set.

[0029] A still further preferred feature of the present invention is to manage HTTP connection requests from a client machine using a so-called virtual URL that defines a user session. The client is then notified that that the requested URL has been moved to a specific server.

[0030] Yet another preferred feature of the present invention is to balance connection requests to a pool of servers on a session basis so that each server has a predefined number of users that may obtain access to documents supported on that server.

[0031] It is still another preferred feature of this invention to redirect HTTP connection requests originating from a user of a given client machine to a specific server during a session.

[0032] These and other preferred features of the invention are provided in a method, computer program product and server for use managing connection requests to a pool of servers identified by a given URL. In one embodiment, the method begins in response to a connection request from a given client machine that initiates a user session. In response to the request, the method associates a session identifier with a given server in the pool. The session identifier is then used to generate a "virtual" URL that redirects the connection request to the given server. Thereafter, any additional connection requests issued from the given client machine during the user session are redirected to the given server so that all content is served to the client from the same location. When the user session terminates, the virtual URL is inactivated and the given server is returned to the pool so that it can then be assigned a new user session to manage.

[0033] Thus, according to preferred embodiments of the invention, a method for managing connection requests to a pool of servers is responsive to connection requests from client machines that initiate user sessions for associating each user session originating from a client machine with a given server in the pool. User sessions are associated with the servers in the pool in accordance with a load balancing protocol. During each user session, any additional connection requests originating from a given client machine are then redirected to the server that is managing the session. The user

session is associated with a given server by generating a virtual URL that includes a unique session identifier, which identifier is then provided with all data returned from the server to the client machine. The virtual URL is inactivated upon completion of the user session.

[0034] Preferred embodiments of the invention also preferably comprise a server for managing a pool of servers at a Web site identified by a given URL. The server may comprise a processor, an operating system, a load balancing routine, and a redirector routine for managing HTTP connection requests to the Web site. The redirector routine is a front end process that runs in the server and includes means responsive to connection requests from client machines that initiate user sessions for associating each user session originating from a client machine with a given server through a unique session identifier. During each user session, connection requests originating from the client machine to the given URL are redirected to the given server.

[0035] A preferred embodiment of the present invention will now be described by way of example, with reference to the accompanying drawings, in which:

Figure 1 is a representative system in which a preferred embodiment of the present invention is implemented;

Figure 2 is a flowchart illustrating the conventional processing associated with an HTTP request from the Web client to the Web server shown in Figure 1;

Figure 3 is a block diagram of a client machine connecting to a Web site comprising a server pool;

Figure 4 is a state diagram illustrating the inventive functionality of the redirector routine; and

Figure 5 is a flowchart illustrating the function of the redirector routine.

[0036] A known Internet client-server system is implemented is illustrated in Figure 1. A client machine 10 is connected to a Web server 12 via network 14. For illustrative purposes, network 14 is the Internet, an intranet, an extranet or any other known network. Web server 12 is one of a plurality of servers which are accessible by clients, one of which is illustrated by machine 10. A representative client machine includes a browser 16, which is a known software tool used to access the servers of the network. The Web server supports files (collectively referred to as a "Web" site) in the form of hypertext documents and objects. In the Internet paradigm, a network path to a server is identified by a so-called Uniform Resource Locator (URL).

[0037] A representative Web server 12 is an IBM Netfinity server comprising a RISC-based processor 18, the AIX operating system 20 and a Web server program 22, such as Netscape Enterprise Server. The server 12 also includes a display 24 supporting a graphical user interface (GUI) for management and administration, and an Application Programming Interface (API) 23 that provides extensions to enable application developers to extend and/or customize the core functionality thereof through software programs including Common Gateway Interface (CGI) programs, plug-ins, servlets, active server pages, server side include (SSI) functions or the like.

[0038] A representative Web client is a personal computer that is x86-, PowerPC- or RISC-based, that includes an operating system such as IBM OS/2 or Microsoft Windows '95, and that includes a Web browser, such as Netscape Navigator 4.0 (or higher), having a Java Virtual Machine (JVM) and support for application plug-ins or helper applications.

[0039] The Web server accepts a client request and returns a response. The operation of the server program 22 is governed by a number of server application functions (SAFs), each of which is configured to execute in a certain step of a sequence. This sequence, illustrated in Figure 2 by way of background only, begins with authorization translation (AuthTrans) 30, during which the server translates any authorization information sent by the client into a user and a group. If necessary, the AuthTrans step may decode a message to get the actual client request. At step 32, called name translation (NameTrans), the URL associated with the request may be kept intact or it can be translated into a system-dependent file name, a redirection URL or a mirror site URL. At step 34, called path checks (PathCheck), the server performs various tests on the resulting path to ensure that the given client may retrieve the document. At step 36, sometimes referred to as object types (ObjectType), MIME (Multipurpose Internet Mail Extension) type information (e.g., text/html, image/gif, etc.) for the given document is identified. At step 38, called Service (Service), the Web server routine selects an internal server function to send the result back to the client. This function can run the normal server service routine (to return a file), some other server function (such as a program to return a custom document) or a CGI program. At step 40, called Add Log (AddLog), information about the transaction is recorded.

[0040] Figure 3 is a block diagram illustrating the environment in which the present invention is implemented. In this illustration, client machine 42 makes HTTP requests to a Web site comprising a managing server 44, and a set of mirrored servers 46a-46n managed by the managing server. Managing server 44 may comprise one of the mirrored servers. Managing server 44 includes a load balancing routine 48 for keeping track of the load on each of the mirrored servers and for directing service requests to the servers according to a load balancing algorithm. According to the present invention, the managing server further includes a redirector routine 50 that provides the enhanced functionality of the present invention.

[0041] As will be seen, redirector routine 50 acts as a front end process for parsing client requests and for determining whether a given request represents the beginning or end of a "session." As used herein, a session represents a set of connection-less transactions between a given user (at a client machine) and the Web site. For example, if the managing server is a bank Web site, the session involves a set of queries to the server from the user, together with the responses served from the servers. In this example, a user logs in and is recognized as being authorized to access given information (e.g., a bank account balance). According to the present invention, the redirector enables all of requests originating from the client machine to the site, and all of the information delivered to the client machine, to be served to and from the same server during a given "session." One of ordinary skill in the art will thus appreciate that this enables the load to be balanced on a per-session basis, as opposed to a per-connection basis as in the prior art.

[0042] This operation is now described with respect to the state diagram of Figure 4, together with the flowchart of Figure 5. Sequential steps are numbered in the state diagram. The routine begins at step 52 with the client making a request to initiate a session. Typically, this is a login request, which may occur, for example, by having the user access the Web site (through its URL) and then enter information into a CGI-based form. This is a conventional login transaction over the Internet. Returning to the flowchart, in step 52, the client machine thus issues an HTTP request (for example, http://www.bank.com/login.html) to the managing server. At step 54, the redirector intercepts the request and recognizes that the user has requested the initiation of a session. Following (optional) entry and validation of a userid and password, the routine continues at step 56, wherein the redirector queries the load balancing routine to determine which of the servers in the set will service the session. This outcome of this test depends on the particular load balancing algorithm being implemented. As noted above, the present invention enables HTTP connection requests and associated server responses to be managing through one server of the set throughout the entire session.

[0043] At step 58, the redirector associates a session identifier with the session and then notifies the server (in this example, server S4) that has been selected to manage the session of the session identifier. Preferably, the session identifier is a unique number (e.g., an increasing sequential number). At step 60, server S4 records the session identifier in a session table 55. Each server in the server pool preferably has an associated session table 55 for storing session identifiers of the sessions being managed by that server. The redirector likewise includes an appropriate data structure 57 maintaining information about which server is managing which session. At step 62, the redirector returns an appropriate redirection response (e.g., "302: URL has moved") to the client. This response also identifies the URL of the server and includes the session identifier, e.g.: "http://server 4 URL/session identifier/login.html. This is sometimes referred to herein as a "virtual" URL. This completes the redirection function.

[0044] The client then continues at step 64 by issuing a new HTTP connection request, using the URL passed from the managing server during step 62. This redirection process is typically an automatic function carried out by the browser in response to the 302 response. The routine then continues at step 66 with the server S4 serving the requested data. All data returned to the client contains the session identifier as part of the URL base. Throughout the remainder of the session, all HTTP connection requests from this particular client machine are managed through server S4. This is illustrated at step 68 in the flowchart.

[0045] A test is performed repeatedly at step 70 to determine whether the session is to be terminated. Typically, this request is identified by the redirector because the user has taken some action that generates a given logoff request. Thus, for example, the redirector has parsed the input HTML stream and recognized the following request: "http://server 4 URL/session identifier/logoff.html". At this point in the routine, namely, at step 72, the server S4 deletes the session identifier from the session table 55 and performs the requested logoff action. Deletion of the session identifier is sometimes referred to as "inactivating" the identifier because, once the session identifier is removed from the table, the client machine is no longer explicitly coupled to the server that, up until that time, had been handling the connection requests originating from that machine during the user session.

[0046] The routine then continues at step 74 to return a logoff screen to the client machine. The server then notifies the redirector that the user associated with the session identifier has logged off. At step 76, the redirector releases the server S4 from the session, thus making the server available to handle a new session from another user.

[0047] Thus, the redirector maintains a running count of the number of sessions being managed by each server in the pool, as well as the identification of the actual session identifiers being managed at each server. As a user completes his or her session, the server that has been managing the session becomes available to the redirector (and, thus, the load balancing routine) to service another set of connection requests from another user. Thus, load balancing is achieved on a per-session, as opposed to a per-connection, basis. This provides significant advantages over the prior art.

[0048] In particular, a given user may now undertake a set of transactions and be assured that all such transactions are managed by the same server. By implementing the redirection function, the user is assured that he or she may carry out a set of transactions without having to login repeatedly to the server.

This is quite advantageous as the transactions are connection requests that are communicated in a connection-less (i.e. a "stateless") operating environment. By adding the virtual URL to the redirection request, the target server has the capability to count actual user sessions and thus impose an upper limit of users instead of connections. Another benefit is that if the user cannot bookmark a page in the middle of a session because the virtual URL is short-lived (i.e. it only lasts as long as the session lasts). This ensures that the user is forced to back to the beginning URL to begin a new session, thus preventing restarts in the middle of a particular session. The redirector provides a consistent session flow.

[0049] The above-described functionality preferably is implemented in software running on the managing server. There is no requirement to modify the client-side software as the redirection to the virtual URL takes place automatically. The redirector thus may be plug-in enabled code, preferably a Java servlet. The functionality thus is implemented in software executable in a processor, namely, as a set of instructions (program code) in a code module resident in the random access memory of the computer. Until required by the computer, the set of instructions may be stored in another computer memory, for example, in a hard disk drive, or in a removable memory such as an optical disk (for eventual use in a CD ROM) or floppy disk (for eventual use in a floppy disk drive), or downloaded via the Internet or other computer network.

[0050] In addition, although the various methods described are conveniently implemented in a general purpose computer selectively activated or reconfigured by software, one of ordinary skill in the art would also recognize that such methods may be carried out in hardware, in firmware, or in more specialized apparatus constructed to perform the required method steps.

[0051] Further, as used herein, a Web "client" means any computer or component thereof directly or indirectly connected or connectable in any known or later-developed manner to a computer network, such as the Internet. The term Web "server" means a computer, computer platform, an adjunct to a computer or platform, or any component thereof. Of course, a "client" means one who requests or gets the file, and "server" is the entity which downloads the file.

Claims

1. A method for managing connection requests to a pool of servers identified by a given URL, comprising the steps of:

   in response to a connection request from a given client machine that initiates a session, associating a session identifier with a given server in the pool;

   using the session identifier to redirect the connection request to the given server; and

   during the session, redirecting to the given server any additional connection requests from the given client machine.

2. The method as claimed in claim 1 wherein the step of using the session identifier includes generating a virtual URL.

3. The method as claimed in claim 2 wherein the virtual URL comprises a URL in the connection request modified to include the session identifier.

4. The method as claimed in any preceding claim wherein the session identifier is incorporated in data returned from the given server to the client machine.

5. The method as claimed in any preceding claim further including the step of:

   in response to a connection request from the given client machine that terminates the session, inactivating the session identifier.

6. The method as claimed in any preceding claim wherein the given client machine includes a browser.

7. The method as claimed in any preceding claim wherein each of the servers in the pool supports a similar set of objects.

8. The method as claimed in any preceding claim wherein the session identifier is associated with a given server as a function of a load balancing protocol.

9. A computer program product in a computer-readable medium for managing connections requests to a pool of servers, comprising:

   means responsive to connection requests from client machines that initiate user sessions for associating each user session originating from a client machine with a given server in the pool in accordance with a load balancing protocol; and

   means operative during each user session for redirecting to the given server any additional connection requests originating from the client machine.

10. A server for managing a pool of servers at a Web site identified by a given URL, comprising:

a processor;

an operating system;

a load balancing routine; and

a redirector routine for managing HTTP connection requests to the Web site, comprising:

means responsive to connection requests from client machines that initiate user sessions for associating each user session originating from a client machine with a given server in the pool in accordance with the load balancing routine; and

means operative during each user session for redirecting to the given server any additional connection requests originating from the client machine.

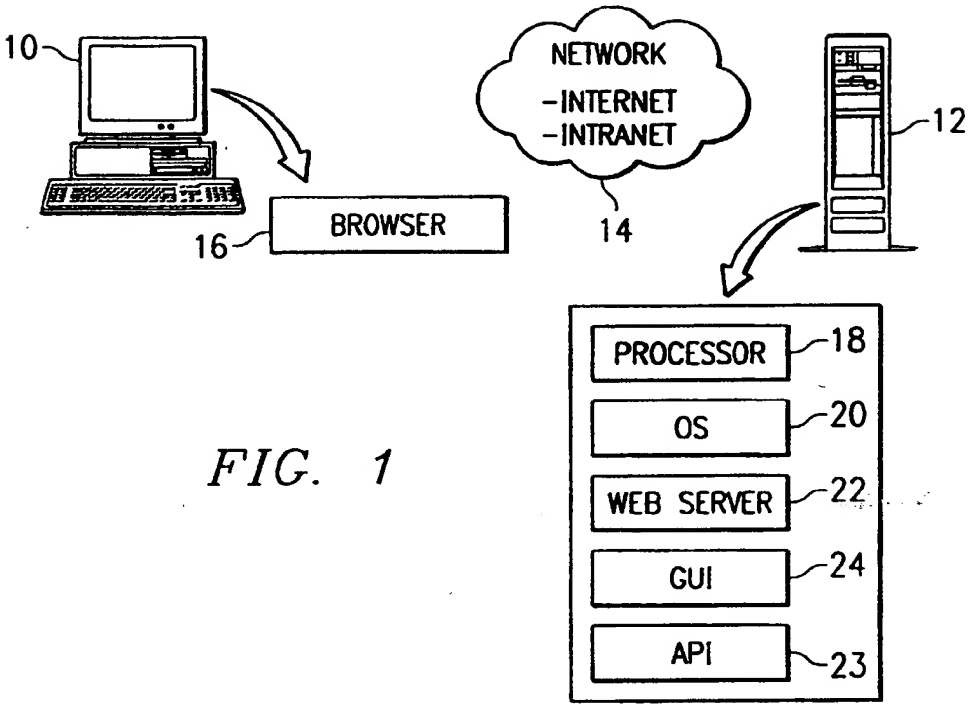

FIG. 1

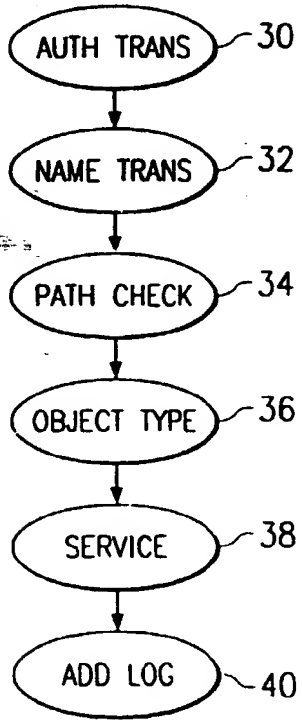

FIG. 2

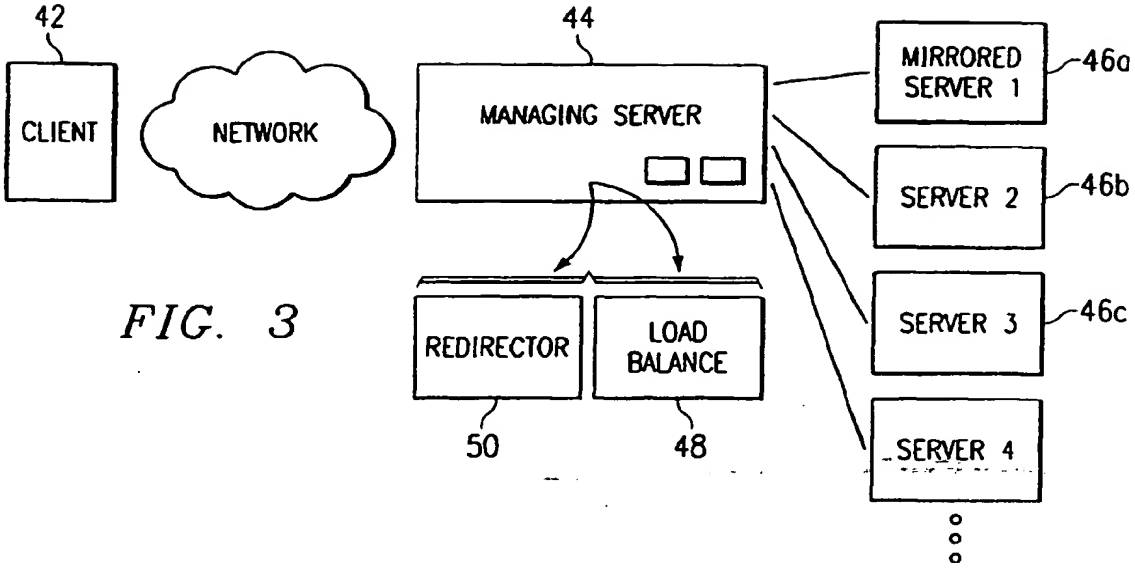

FIG. 3

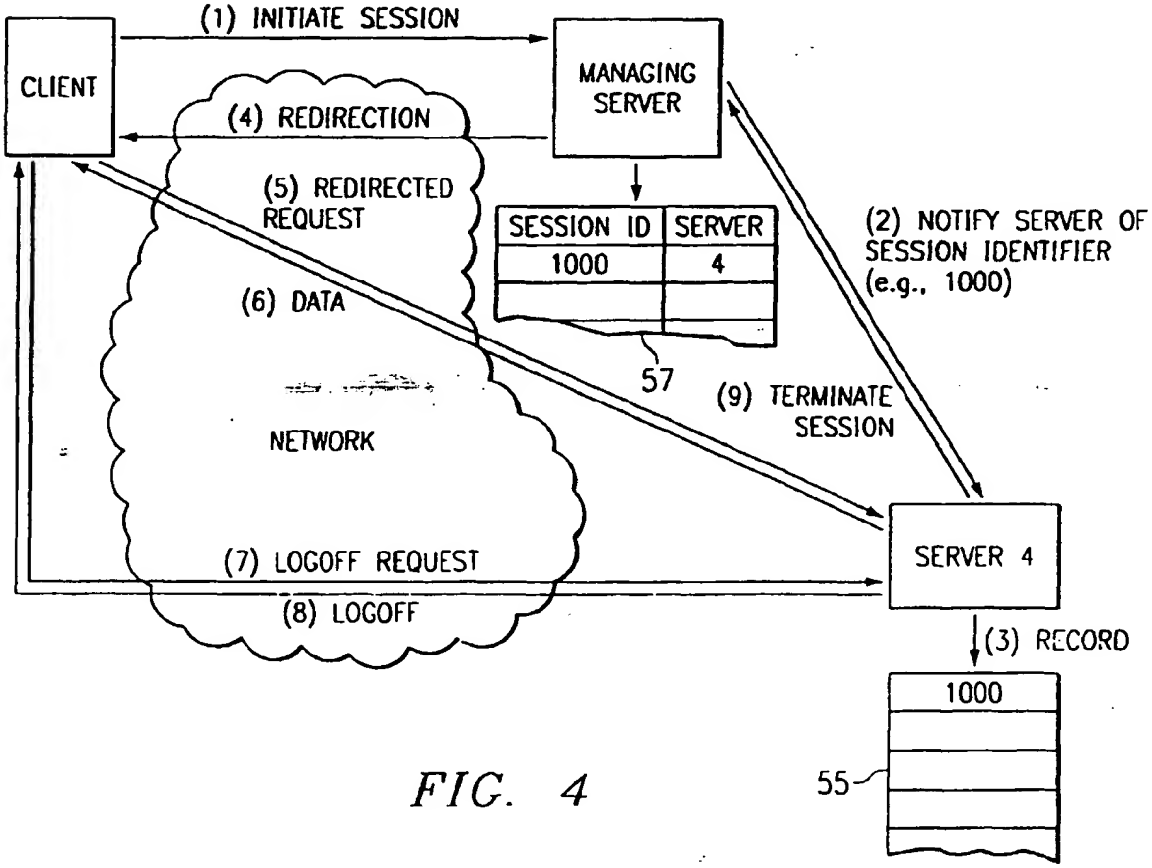

FIG. 4

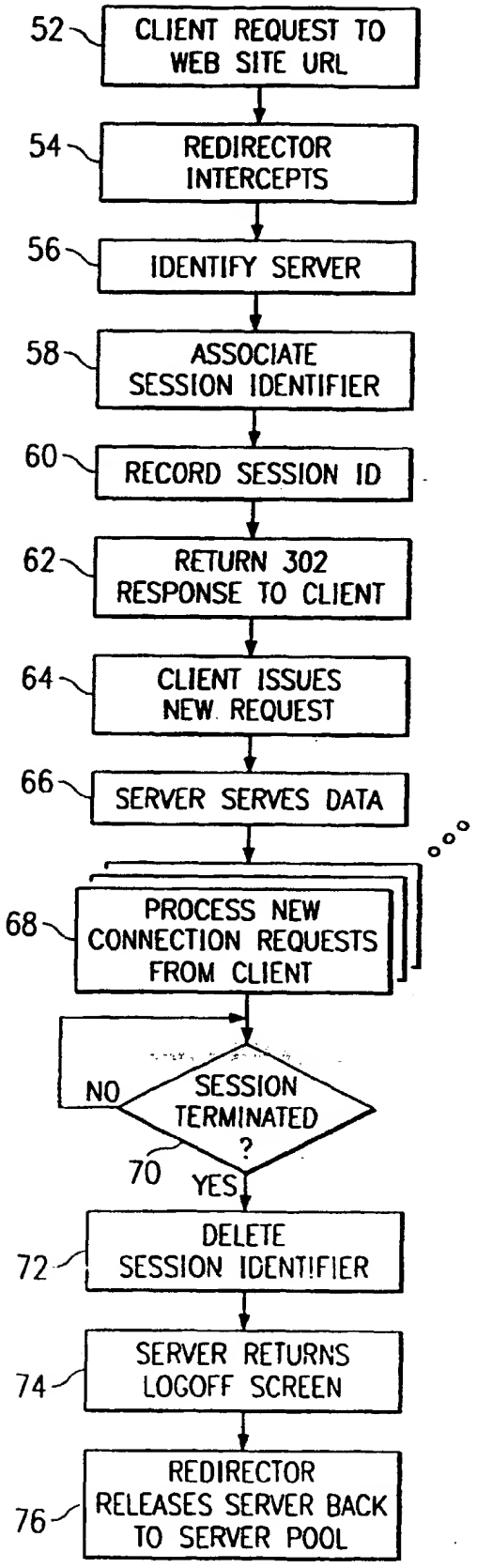

FIG. 5

(19) Europäisch s Patentamt
Europ an Patent Office
Office eur péen des brevets

(11) EP 1 041 496 A3

(12) EUROPEAN PATENT APPLICATION

(88) Date of publication A3:
16.01.2002 Bulletin 2002/03

(43) Date of publication A2:
04.10.2000 Bulletin 2000/40

(21) Application number: **00302491.6**

(22) Date of filing: **27.03.2000**

(51) Int Cl.[7]: **G06F 17/30**, H04L 29/06

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: **31.03.1999 US 282692**

(71) Applicant: **International Business Machines Corporation**
Armonk, NY 10504 (US)

(72) Inventor: **Lita, Christian,**
c/o IBM United Kingdom Ltd.
Winchester, Hampshire SO21 2JN (GB)

(74) Representative: **Burt, Roger James, Dr.**
IBM United Kingdom Limited Intellectual Property Department Hursley Park
Winchester Hampshire SO21 2JN (GB)

(54) Using virtual URLs for load balancing

(57) A method, computer program product and server for use managing connection requests to a pool of servers identified by a given URL. The method begins in response to a connection request from a given client machine that initiates a user session for associating a session identifier with a given server in the pool. The session identifier is then used to generate a "virtual" URL that redirects the connection request to the given server. Thereafter, any additional connection requests issued from the given client machine during the user session are redirected to the given server so that all content is served to the client from the same location. When the user session terminates, the virtual URL is inactivated and the given server is returned to the pool so that it can then be assigned a new user session to manage.

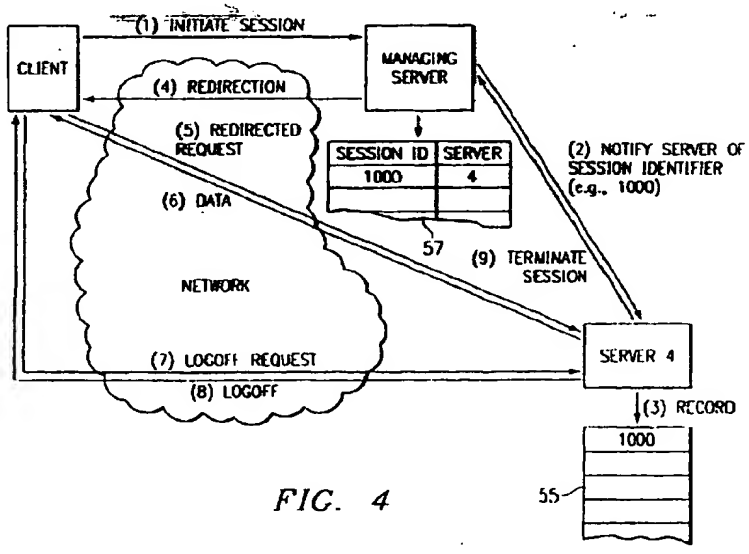

FIG. 4

EP 1 041 496 A3

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 00 30 2491

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
X	US 5 774 668 A (CHOQUIER PHILIPPE ET AL) 30 June 1998 (1998-06-30) * column 1, line 15 - line 23 * * column 1, line 64 - column 2, line 52 * * column 7, line 64 - column 8, line 13 * * column 8, line 65 - column 9, line 16 * * column 13, line 17 - line 23 * * figure 5A *	1-10	G06F17/30 H04L29/06
A	ARUN IYENGAR: "Dynamic Argument Embedding: Preserving State on the World Wide Web" IEEE INTERNET COMPUTING, IEEE SERVICE CENTER, PISCATAWAY, NJ, US, 1 March 1997 (1997-03-01), pages 50-56, XP002164484 ISSN: 1089-7801 * whole document * * p. 55, paragraph "implementation" - p. 56, paragraph "conclusion" *	1-10	
A	WO 98 26359 A (WALL DATA INC) 18 June 1998 (1998-06-18) * page 3, line 11 - line 26 * * page 4, line 25 - line 33 * * page 28, line 1 - page 30, line 11 * * figure 14 *	1-10	TECHNICAL FIELDS SEARCHED (Int.Cl.7) H04L G06F
A	MOURAD A ET AL: "SCALABLE WEB SERVER ARCHITECTURES" PROCEEDINGS IEEE INTERNATIONAL SYMPOSIUM ON COMPUTERS AND COMMUNICATIONS, XX, XX, 1 July 1997 (1997-07-01), pages 12-16, XP000199852 * abstract * * whole section 1 * * whole section 2.1 * * whole section 4.5 *	1,6-10	

-/--

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
MUNICH	23 November 2001	Wienold, N

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 00 30 2491

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
A	US 5 774 660 A (LIU ZAIDE ET AL) 30 June 1998 (1998-06-30) * abstract * * column 6, line 8 - column 7, line 30; figures 7,10,11A,11B * * column 21, line 1 - line 19 * * claims 1-6,12 * -----	1-10	

TECHNICAL FIELDS SEARCHED (Int.Cl.7)

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
MUNICH	23 November 2001	Wienold, N

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 00 30 2491

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report
The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

23-11-2001

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 5774668 A	30-06-1998	US 5951694 A	14-09-1999
WO 9826359 A	18-06-1998	US 5968119 A	19-10-1999
		AU 4352297 A	03-07-1998
		EP 0960377 A1	01-12-1999
		WO 9826359 A1	18-06-1998
		US 6081837 A	27-06-2000
		US 6122661 A	19-09-2000
US 5774660 A	30-06-1998	US 6182139 B1	30-01-2001

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82